UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/627,413	07/27/2000	Theron D. Tock	9803-0099-999	1174

32291 7590 04/26/2004

MARTINE & PENILLA, LLP
710 LAKEWAY DRIVE
SUITE 170
SUNNYVALE, CA 94085

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	8

DATE MAILED: 04/26/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/627,413	TOCK ET AL.	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 January 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *22-51(claims 1-21 being canceled)* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *22-51* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 1/26/2004.

As indicated in Applicant's response, claims 22, 26, 28, 32, 36, 38, 42, 46, and 48 have been amended. Claims 22-51 are pending in the office action.

The terminal disclaimer filed 1/26/04 is also considered and the double patenting rejection withdrawn.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 22-51 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hastings, USPN: 5,193,180 (hereinafter Hastings), in view of Gupta, USPN: 5,822,590 (hereinafter Gupta).

As per claim 22, Hastings discloses a method of generating object-oriented programs for accessing and updating memory stored objects, the method comprising:

receiving an initial program that includes original instructions for accessing objects stored in computer main memory (e.g. *old file.o* – Fig. 1);

scanning the initial program to automatically identify object accessing instructions and corresponding locations at which instructions are to be added representing a first set of identified program locations (e.g. col. 3, lines 29-56; col. 8, line 61 to col. 10, line 21; Fig. 5-8 – Note:

using relocation table, offset readjustment, to insert instructions for memory accesses is equivalent to identifying locations to insert accessing instructions);

automatically revising the initial computer program to generate a revised program by adding object loading instructions to the initial program at the first set of identified locations (e.g. col. 8, line 61 to col. 10, line 21; Fig. 5-8; col. 12, lines 61-67; col. 13, lines 27-50 – Note: loading instructions is inherent to memory accesses instructions; using insertion of code, patching into a precompiled code or linked code is equivalent to automatically generating revised code).

But Hastings does not explicitly disclose accessing and updating persistently stored objects nor does Hastings specify that the added loading instructions, when executed, load respective ones of the objects from persistent storage into the main memory when each respective persistent object is accessed and not already in the main memory. Hastings discloses inserting code for maintaining memory stability (e.g. col. 9, lines 38 to col. 10, line 18) and allocation/de-allocation error preventing (e.g. col. 3, lines 29-56), thus suggesting maintaining consistency between data fetched into and read from memory. The concept of loading data from a non-volatile and more persistent storage medium to be used at runtime into a fast or more volatile memory to provide for potential memory conflicts due to data not available prior to execution was a known concept in the art of memory loading and data fetching at the time the invention was made. In a method to preprocess an initial program to establish memory access consistency using pointer table and offset information analogous to Hastings, Gupta discloses accessing and updating persistently stored objects (Object oriented database), e.g. restoring dereferenced objects to memory, and generating of instructions after preprocessing to

dynamically load objects into memory (e.g. col. 1, line 46 to col. 2, line 45; Fig. 1-2; *inserts code to update dtable with OID* - col. 4, line 24 to col. 5, line 36). It would have been obvious for one of ordinary skill in the art at the time the invention was made to provide the load instructions as taught by Gupta's persistent model language to the set of inserted instructions as suggested by Hastings to access and update persistently stored objects, and to load objects persistent objects into memory because memory accessed objects cannot limit to just program memory allocation and should be persisted in more non-volatile storage in order to alleviate burden on volatile memory, thus efficiently preserve resources that would be used for memory fault checking as intended by Hastings.

Nor does Hastings disclose that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects prior to adding object load instructions to the initial program. However, updating a database implies modification of its inherent data structures or records or tables (e.g. *to update dtable* – col. 5, lines 33-36) and bringing objects from disk into memory implies main copies of persistently stored objects as taught by Gupta. Further, in view of the teachings by Gupta, some pointer structures are re-addressed in order to accommodate to changes related to persistent objects acknowledged in the preprocessing of the initial program (e.g. *dbpointer is made to point at* - col. 4, lines 6-58; col. 6, lines 6-41); and some instructions are already allocated for accessing persistent objects when initially submitted for preprocessing (e.g. col. 2, lines 10-27). In the scenario that persistently stored objects are to be kept consistent with the debug involving data fetched for runtime memory conflict prevention as intended by Hastings, it would have been

obvious for one of ordinary skill in the art at the time the invention was made to allocate persistent objects accessing instructions to Hastings' initial program so that readjustment of such instructions (e.g. modifying of data structures) can be effected in order to provide load instructions necessary to maintain such consistency according to the persistent update scheme taught by Gupta as mentioned above, especially when data coherency (as well-known in the art) between persistent storage and volatile storage would help alleviate further complications when subsequent execution require that data are to be refetched and reloaded.

As per claim 23, Gupta discloses a pointer indicating that the object does not exist in memory (e.g. col. 4, lines 24-38); and the motivation to add the load instructions to respond only when the object to be accessed is referenced by a null indicator as taught by Gupta to Hastings's memory checking instructions has been set forth above for the same rationale.

As per claim 24, Hastings discloses object marking data, or status bits in conjunction with the added instructions to the initial program such as status array marking memory status of the memory and updating of such memory status bits (e.g. col. 9, line 38 to col. 11, line 36); but does not explicitly disclose adding dirty object marking instructions to the program; but the concept of using data structure in conjunction with added instructions to perform indication marking of program objects, i.e. indication a new and/or updated object, in order perform data or memory objects update, memory load or object persisting has been disclosed.

But Hastings does not specify adding object storing instructions to generate a revised instructions operable to store respective objects from memory into persistent storage wherein said respective objects are indicated by the object marking data. In view of the teachings by Gupta to use external structure information to resolve memory object consistency and to insert

instructions to persist object from memory to non-volatile storage (e.g. col. 5, lines 10-36), it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide to Hastings' set of added instructions (based on the above marking technique) those that would dynamically store memory persistable objects as taught by Gupta into persistent storage, i.e. object storing instructions, because this would alleviate resources to update the persistent storage that would otherwise be required in another pass for checking in the revised program what persistable objects are to be written to persistent storage.

As per claim 25, Hastings only disclose inserting within instructions in the initial program with instructions based on offset information and relocation tables (re claim 22) but does not specify storing instructions with instructions for replacing certain respective memory objects reference with respective persistent objects identifiers. But Gupta specify generating instructions after a preprocessing to associate persistent object with global data structure for persisting objects via using those instructions (e.g. *OID, dbtable* - col. 3, line 36 to col. 4, line 3). In view of the rationale to combine Hastings and Gupta to provide storing instructions as mentioned above, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide the object ID and instructions to replace the memory persistable objects with respective persistently stored objects thus taught by Gupta in order to enhance the object referencing suggested in Hastings when Hastings uses relocation and offset pointing structure to efficiently track the effect of code insertion and memory consistency checking.

As per claim 26, Hastings discloses a method of generating object-oriented (e.g. col. 4, lines 48-54) programs for accessing and updating memory objects, the method comprising: receiving (for accessing, updating, committing transactions); scanning (for identifying locations

at which to add instructions); revising (to generate revised program); all of these limitations have been addressed in claim 22 above, and are herein rejected using the rationale as set forth in claim 22. But Hastings does not specify accessing and updating persistent objects but this has been addressed in claim 22. Nor does Hastings disclose adding (dirty object marking instructions) and adding (storing instructions); but these limitations have been addressed in claim 24.

Nor does Hastings disclose that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claim 27, this claim corresponds to claim 25, hence is rejected herein using the same rationale as set forth therein.

As per claim 28, Hastings discloses a method comprising: scanning an initial program to automatically identify object accessing and updating instructions and corresponding locations at which additional instructions are to be added (e.g. col. 8, line 61 to col. 10, line 21; col. 10, lines 23-61 - Fig. 3-8 -Note: identifying memory access instructions with memory allocation/de-allocation with write operations implicitly discloses instructions of both for accessing and updating memory, i.e. memory status update, e.g. col. 9, line 64 to col. 12, line 28). The limitation as to access and update persistently stored objects has been addressed using Gupta's teachings.

Further, Hastings discloses the steps:

automatically revising (for adding instructions at identified locations);

adding (set of instructions of object accessing, each object is accessed and not already in memory);

adding (second set of instructions for object updating instruction, each object is updated for a first time). All of these steps have been addressed in claim 22 and further in Hastings: col. 9, line 64 to col. 12, line 28.

But Hastings does not specify that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claim 29, this limitation of loading of objects from persistent storage to main memory has been addressed in the combination as set forth in claim 22 above, using Gupta's teachings.

As per claim 30, this claim includes the limitations to add object-marking instructions indicating new and/or updated data memory objects and has been addressed in claim 24; and is rejected using the rationale set forth therein.

As per claim 31, this claim corresponds to the limitation to commit memory objects to persistent storage; and adding a third set of instructions such as storing instructions as recited in claim 24; hence is rejected herein using the corresponding rejection as set forth in claim 24 (Note: to identify which memory objects that are persistable as taught by Gupta is equivalent to add instructions at corresponding locations to store those objects into persistent storage, or commit to persistent storage).

As per claim 32, Hastings discloses a program product for use in conjunction with a computer having main memory and persistent storage (e.g. Fig. 1 – Note: file system is equivalent to persistent storage; hard disk being an inherent persistent storage)such program comprising:

a postprocessor procedure for modifying an initial program that includes instructions for accessing and updating objects stored in a computer's main memory (e.g. col. 8, line 61 to col. 10, line 21; col. 10, lines 23-61 - Fig. 3-8 -Note: identifying memory access instructions with memory allocation/de-allocation with write operations implicitly discloses instructions of both for accessing and updating memory, i.e. memory status update, e.g. col. 9, line 64 to col. 12, line 28);

said postprocessor procedure including instructions to perform the steps of

receiving (initial computer program);

scanning (for object accessing instructions to be added);

automatically revising (to add load instructions). These steps limitations have been addressed in claim 22 above using the combined teachings of Hastings and Gupta.

But Hastings does not specify that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claim 33, this claim corresponds to claim 23, hence is rejected likewise.

As per claims 34-35, see rejections of claims 24-25 respectively.

As per claim 36, Hastings discloses a program product comprising:

a postprocessor procedure (for modifying – see rejection in claim 32 above);

the postprocessor procedure having instructions for

receiving an initial program that includes instructions for accessing, updating objects stored in the main memory (re claim 32);

scanning the program to automatically identify object updating instructions and corresponding locations at which to add instructions (re claim 32);

automatically revising to generate a revised program (re claim 32).

But Hastings does not specify that the initial program includes committing transactions nor does Hastings disclose scanning to identify committing transactions; whereas Gupta discloses preprocessor identifying of objects that are persistable, hence discloses post-preprocessor program including committing transactions and identifying of locations at which to add transaction committing instructions (e.g. col. 5, lines 10-36 – Note: commit an object to persistent store is equivalent to storing object identified as persistable object to non-volatile storage). The motivation to combine Hastings' teachings as to add instructions to access and update memory objects with the persistent store instructions as taught by Gupta has been set forth in claim 24 above.

Nor does Hastings teach adding dirty object marking instructions and object storing instructions based on the object marking instructions; but this limitation has been addressed in claim 24 above.

Nor does Hastings teach that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating

revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claim 37, refer to claim 25 or 27.

As per claim 38, Hastings discloses a program product comprising a post-processor procedure as addressed in claim 32;

such post-processor procedure including instructions for performing the steps of

scanning (identifying accessing and updating instructions);

automatically revising (generate revised program by adding);

adding first set of instructions (for memory object accessing);

adding a second set of instructions (for memory updating instructions). All those steps have been addressed in claim 28.

Nor does Hastings teach that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claims 39-41, refer to rejections of claims 29-31 respectively.

As per claim 42, Hastings discloses a computer system comprising a central processing unit; memory, including main memory and persistent storage (e.g. Fig. 1 – Note: file system is example of persistent storage), an initial computer program (e.g Fig. 1); and a postprocessor procedure for modifying the initial program so as to generate the revised program, the initial program including instructions for accessing objects stored in main memory (re claim 36 for

corresponding rejection); the postprocessor procedure including instructions for performing the steps of receiving (initial program);

scanning (accessing and object access instructions);

automatically revising (generate revised program);

adding (object loading of instructions). All these steps limitations have been addressed in claim 22 above.

But Hastings does not specify that the added loading instructions when executed, load respective objects from persistent storage into main memory as recited in claim 22. However, this limitation has been addressed therein using Gupta's teachings.

Nor does Hastings disclose that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claims 43-45, refer to claims 23-25, respectively.

As per claims 46, this claim includes the computer system as addressed in claim 42 and further includes a post-processor procedure having instructions for performing the same steps as recited in claim 36; hence is rejected with the corresponding rejection for each steps as set forth in claim 36.

Nor does Hastings disclose that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claim 47, this claim corresponds to claim 37, hence is rejected herein using the same rationale as set forth therein.

As per claim 48, this claim includes the computer system as addressed in claim 42 and further includes a post-processor procedure having instructions for performing the same steps as recited in claim 38; hence is rejected with the corresponding rejection for each steps as set forth in claim 38.

Nor does Hastings disclose that the original instructions includes instructions for accessing persistent objects comprising main memory of persistently stored objects; or disclose generating revised program by modifying data structures of the persistent objects. These limitations have been addressed in claim 22 above.

As per claims 49-51, see rejections of claims 39-41 respectively.

Response to Arguments

4. Applicant's arguments filed 22-51 have been fully considered but they are not persuasive.

(A) Applicants have submitted that Hastings does not teach modifying data structures of the persistent objects (Appl. Rmrks, pg. 16, 3rd para). It is noted that maintaining consistency between data stored in a more persistent basis and data stored in a temporary or volatile basis is a known concept in the art. Hastings teaches a debug environment to provide code readjustment against potential memory conflicts and in the same line of approach, Gupta uses a database for persistently store objects which are to be updated or kept consistent with memory data being fetched or dereferenced during such debug process as analogously intended by Hastings. Gupta's purpose to avert memory reference conflicts added to Hastings debug environment in the light of what has been known in the art would be ample motivation as to why one of ordinary

skill in the art would use Gupta to remedy to the persistently stored objects not mentioned in Hastings' method. In updating persistent objects in the object-oriented database, Gupta has implicitly disclosed modifications of table or records stored therein. The combined teachings have addressed the limitation and have shown why such combination would have been obvious according to the well-known practice of data coherency as mentioned in the rejection.

(B) Applicants have submitted that Hastings does not provide modification related to the data structure of any existing object or data structure used by the original program and that there is no distinction between persistent objects accessing and memory data accessing (Appl. Rmrks, pg. 16, last para to pg. 17, top 2 para). Modifying data structures used by the original program, Hastings does disclose just that when offsets are readjusted or expansion program instructions are inserted in a code(Fig. 4-10). The claim does not provide details as to distinguish program data structures in the initial program from what is disclosed by Hastings. As far as structures related to persistent objects, Gupta is brought in to provide the grounds as to why modifications to the original code is necessary to provide code/data allocation to avert memory conflict during debug so as to maintain coherency with objects persistently stored and for which code allocating instructions are provided for preprocessing (see rejection and section A above). Applicants have failed to show how as combined Gupta and Hastings cannot fulfill the requirements of the claim and how such combination would lead to inapposite and conflicting results.

(C) Applicants have argued that Hastings' attempts to access memory data that does not contain any data is contrary to the invention such that 'the revised program ... into main memory the first time ... so to avoid generating any error message' (Appl. Rmrks, pg. 17, middle para). Applicants have admitted that the fact of loading data in memory is to avoid an error; and

Hastings is providing just that by loading data or allocating data into memory to avoid memory conflicts (e.g. col. 9, line 38 to col. 10, line 21), the concept of being accessed for memory usage the first time is implied when an object is being loaded or allocated for such context.

(D) Applicants have submitted that Gupta does not teach 'modifying data structures of the persistent objects and that global data shared in memory is not the object-oriented programming privacy of one object (Appl. Rmrks, pg. 17, 4th para). Gupta teaches a object-oriented database and inserting or creating instructions upon detecting instructions in the initial code referring to a persistent object being susceptible of change (see cols. 2-5 and rejection); and such teachings have met the above raised issues so that when combined with Hastings, the combination would have been obvious as set forth in claim 22 and explained in section A from above. Besides, in the invention background (pg. 1-3), Applicants have admitted that object-oriented programming or OO dBM can be implemented.

(E) Applicants have argued that the 2 subsets of instructions are not disclosed by either Hastings or Gupta (Appl. Rmrks, pg. 18, 2nd para). In response, it is noted that the way Hastings proceeds amounts to detecting where to provide memory data (first set)and where to put data to (second set); and Gupta preprocessing for identifying where to modify the persistent objects to maintain consistency also amounts to a phase to detect (first set) and a phase (second set) to put data to memory or to write to disk. Both methods comprise a first stage of identifying where to bring data to memory and a second phase to actually do the actual change. For the sake of arguments, even if Hastings does not provide the splitting of instructions into 2 sets, Gupta would have provided such 2 sets because Gupta has generated initial instructions so as to enabling when and where to add instructions for persistent objects update.

(D) As for the limitation of a postprocessor (Appl. Rmrks, pg. 18, 3rd and 4th para), the rejection has pointed how the fact to preprocess an initial code by Gupta or Hastings for example, would read on the limitation to have a post processing element to provide code adjustments to enabling memory data loading or persistent data update.

In view of the above remarks, the claims stand as rejected.

Conclusion

5. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please consult Examiner before using this number)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
March 30, 2004

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100